

Project Report

Smart Lender - Applicant Credibility Prediction for Loan Approval

1. INTRODUCTION

1.1 Project Overview

This project focused on the prediction of loan approval of the applicant using machine learning techniques. The prime objective in banking environment is to invest their assets in safe hands where it is. We need to predict whether assigning the loan to particular person will be safe or not. The process involved in Sanctioning loans to the customer is carried out in this project.

- Understanding the Dataset
- Data Pre-Processing
- Comparing ML Algorithms
- Model Building
- UI Design Using Flask
- Prediction

1.2 Purpose

The process of bank credit risk evaluation is recognized at banks across the globe. As we know credit risk evaluation is very crucial, there is a variety of techniques are used for risk evaluation. The prediction of credit defaulters is one of the difficult tasks for many banks. But by forecasting the loan defaulters, the banks definitely may reduce their loss by reducing their non-profit assets. Through this prediction the bank assets will be in safe hands. This prediction system reduces the risk factor behind selecting the safe person so as to save lots of bank efforts and assets.

2. LITERATURE SURVEY

2.1 Existing problem

1. LOAN PREDICTION USING ENSEMBLE TECHNIQUE (Anchal Goyal , Ranpreet Kaur)

The dataset with features, namely, gender, marital status, education, number of dependents, employment status, income, co applicant's income, loan amount, loan tenure, credit history, existing loan status, and property area, are used for determining the loan eligibility regarding the loan sanctioning process. Various ML models adopted in the present method includes, Linear model, Decision Tree (DT), Neural Network (NN), Random Forest (RF), SVM, Extreme learning machines, Model tree, Multivariate Adaptive Regression Splines, Bagged Cart Model, NB and TGA. Ensemble Model gives the better prediction than the individual models. This model also enhances the performance and accuracy of the model. Demerits : Through Ensemble model we compare the several models and choose the best model for our data that helps the organization to make the right decision for the loan request of the costumer. The art of ensembling is hard to learn and any wrong selection can lead to lower predictive accuracy than an individual model. Ensembling is expensive in terms of both time and space. Hence ROI can increase with ensembling

2. LOAN SANCTIONING PREDICTION SYSTEM (Aditi Kacheria, Nidhi Shivakumar, Shreya Sawkar, Archana Gupta)

Loan sanctioning prediction procedure based on NB approach integrated with K-Nearest Neighbor (KNN) and binning algorithms. The seven parameters considered were income, age, profession, existing loan with its tenure, amount and approval status. The sub-processes include, Pre-processing (handling the missing values with KNN and data refinement using binning algorithm), Classification using NB approach and Updating the dataset frequently results in appropriate improvement in the loan prediction process. Missing data can be dealt with K-NN algorithm. K-NN is a simple algorithm that stores all available data and classifies new data based on a similarity measure (e.g., distance functions). The binning algorithm is used for removal of these anomalies. These algorithms will improve the efficiency and make the data set more consistent. To improve the accuracy of the system, a hybrid of Naïve Bayes and K-means can be used. Some demerits are KNN doesn't work well with a large dataset, dataset with a high number of dimensions and it is sensitive to outliers and missing values. Binning leads to loss of information. Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

3. DEVELOPING PREDICTION MODEL OF LOAN RISK IN BANKS USING DATA MINING (Aboobyda Jafar Hamid, Tarig Mohammed Ahmed)

Loan risk prediction model based on the data mining techniques, such as Decision Tree (J48), Naïve Bayes (NB) and BayseNet approaches. The procedure followed was training set preparation, building the model, Applying the model and finally. Evaluating the accuracy. This approach was implemented using Weka Tool and considered a dataset with eight attributes, namely, gender, job, age, credit amount, credit history, purpose, housing, and class. Merits : Three algorithms - j48, bayesNet and naiveBayes algorithms was used to build a predictive models that can be used to predict and classify the applications of loans that introduced by the customers to good or bad loan by investigate customer behaviors and previous pay back credit. Demerits : Decision tree often involves higher time to train the model. Decision tree training is relatively expensive as the complexity and time has taken are more.

4. SURVEY ON PREDICTION OF LOAN APPROVAL USING MACHINE LEARNING TECHNIQUES (Ambika, Santosh Biradar)

The enhancement in the banking sector lots of people are applying for bank loans but the bank has its limited assets which it has to grant to limited people only. The main objective is to predict whether assigning the loan to particular person will be safe or not. Through this proposed model, we can able to predict whether assigning the loan to particular person will be safe or not which will be a safer option for the bank is a typical process and also reducing this risk factor behind selecting the safe person so as to save lots of bank efforts and assets. Merits : Machine learning helps to understand the factors which affect the specific outcomes most. This is done by mining the Big Data of the previous records of the people to whom the loan was granted before. Demerits : The model has to be trained periodically with new training datasets. There were multiple malfunctions in the computers, content errors and fixing of weight in computerized prediction systems.

2.2 References

1. *Prediction of Loan Risk using Naive Bayes and Support Vector Machine* Vimala S.I, Sharmili K.C.2 *International Conference on Advancements in Computing Technologies - ICACT 2018*
2. *Loan Sanctioning Prediction System* Aditi Kacheria, Nidhi Shivakumar, Shreya Sawkar, Archana Gupta *International Journal of Soft Computing and Engineering (IJSCE) 2016*
3. *A survey on Ensemble Model for Loan Prediction* Anchal Goyal , Ranpreet Kaur. *International Journal of Engineering Trends and Applications (IJETA)2016*
4. *Developing Prediction Model Of Loan Risk In Banks Using Data Mining* Aboobyda Jafar Hamid and Tarig Mohammed Ahmed *Machine Learning and Applications: An International Journal (MLAIJ) 2016*

2.3 Problem Statement Definition

The process of bank credit risk evaluation is recognized at banks across the globe. “As we know credit risk evaluation is very crucial, there is a variety of techniques are used for risk level calculation. The prediction of credit defaulters is one of the difficult tasks for any bank. But by forecasting the loan defaulters, the banks definitely may reduce their loss by reducing their non-profit assets

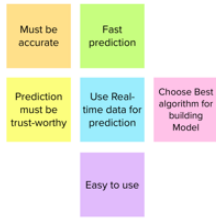
Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Student	Get an education loan	I am not sure about my loan approval	I am unemployed	Upset
PS-2	Businessman	Avail a loan for certain investment	I am not able to get loan on time	Our company's yearly turnover is not up to the mark	frustrated



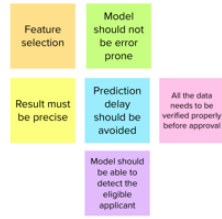
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

Hariharan C



Praveen S



Balaji T



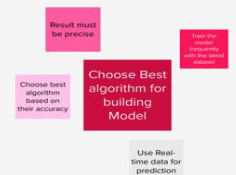
Anandha Kannan N



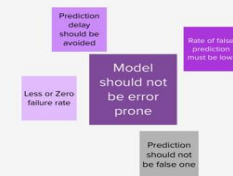
Qualities



Accuracy













Error mitigation



3.2 Ideation & Brainstorming

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To predict the applicant is eligibility for the loan approval. The prediction of credit defaulters is one of the difficult tasks for any bank. But by forecasting the loan defaulters, the banks definitely may reduce their loss by reducing their non-profit assets, so that recovery of approved loans can take place without any loss and it can play as the contributing parameter of the bank statement.
2.	Idea / Solution description	People's Need Increased, so demand for loans in banks also increased. Loan approval is a time consumption process, in order to reduce the time consumption we are going to create a chatbot with audio facility. The loan approval can be predicted using any of the following machine learning algorithms like Linear Regression, Decision Tree, Random Forest. Since the prediction is based on a machine learning algorithm, the loan approval process can be accurate.
3.	Novelty / Uniqueness	Chatbox options will be available, so that the applicant can clarify their queries. Our system will be responding in a faster manner to customer queries.
4.	Social Impact / Customer Satisfaction	Loan approval prediction is a time consuming process the applicant needs to wait for a long time in queue in most of the banks, so with the help of the chatbot option it is easy to identify whether the applicant is eligible or not.
5.	Business Model (Revenue Model)	If the loan approval process time consumption is less, more loan prediction can be easily done, it will help for the bank to raise their business growth.
6.	Scalability of the Solution	Every business sector and normal people are able to use this system.

3.3 Proposed Solution fit

Define CS, fit into CC	1. Customer Segment(s)  Customer is the person who borrow loan for their needs.	6. Customer Constraints  Loan may be available for the customer in their preferred Bank if their documents are in proper manner.	5. Available Solution  Prediction using Machine Learning algorithms like Linear regression, Decision tree algorithm and random forest regression.	Explore AS, differentiate
	2. Jobs-To-Be-Done / Problems  Lenders are often confused for choosing loans, like whether the borrower are eligible are not. This website will help them Predicting eligibility.	9. Problem Root Cause  The root cause of the problem is not having proper documents for borrower and they might enter the incorrect data so that the lender don't have clarity to choose borrower.	7. Behaviour  Loan approval prediction is a time consuming process the applicant needs to wait for a long time, so with the help of chatbot option it is easy to identify whether the applicant is eligible or not.	
Identify strong TR & EM	3. Triggers  Hearing about the website through Advertisements and social media.	10. Your solution  People's needs increased, so demand for loans in banks also increased. Loan approval is a time consumption process, in order to reduce the time consumption we are going to create a chatbot option. The loan approval can be predicted using any of the following machine learning algorithms like Linear Regression, Decision Tree Algorithm and Random Forest Regression.	8. Channels of Behaviour  8.1 Online Moneyfor_u and abhiloans.com helps to explore lending predictions available.	Identify strong TR & EM
	4. Emotions: Before / After  Before: Stress, Hopeless. After: Clarity, Time Saving.		8.2 Offline Ask friends for references in getting a loan.	

3.4 Problem Solution

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The system is easy to use. The interface provided to the user should be simple and clear.
NFR-2	Security	The data provided by the user is highly secured and it should not be misused.
NFR-3	Reliability	The system should be highly reliable as it withstands high load and functions without any crashing.
NFR-4	Performance	The performance of this application is much faster at the same time the result provided by the application is highly accurate.
NFR-5	Availability	This application is available across internet over 24*7, all banks, financial institutions will be making use of our application.
NFR-6	Scalability	The application is highly scalable, because it can run across various operating systems.

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Interaction	User is allowed to view the home page of our website at the initial stage of the process.
FR-2	User Input	User enter all the details on the necessary fields.
FR-3	Verifying the Data	All the data entered in the fields must be verified properly.
FR-4	Retrieving the Data	The application retrieves the values present in fields for further predictions.
FR-5	Predicting the Eligibility	The prediction is done using the most accurate model.
FR-6	Display the Output	The prediction is displayed to the user on the output screen.

4.2 Non-Functional requirements

5. PROJECT DESIGN

5.1 Data Flow Diagrams

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Forms	USN – 1	As a user, I should enter the required data in the forms for prediction	Submit the required data for prediction	High	Sprint – 1
	Model building	USN – 2	I need to build an ML model for Credibility Prediction	I can use the ML model to classify the Credit defaulters	High	Sprint – 2
	User Interface building	USN – 3	As a User, I need a interface to enter my data which acts as a frontend	I can use the HTML CSS webpage which uses Flask to give input to model for prediction	Medium	Sprint – 3
	Prediction	USN – 4	As I have user given data loan credibility check is made by the model.	Pre-processing is done and sent to the model for prediction	High	Sprint – 3
	Deployment of the Webpage in Cloud	USN – 5	As a user, I require to access the webpage anytime everywhere	I can get to the Webpage using the provided Web address	Medium	Sprint – 4
	Deployment of AI model in the cloud	USN – 6	Model would be running on the Cloud	I can access the model anytime everywhere by deploying model in the IBM WATSON	Medium	Sprint – 4

5.2 Solution & Technical Architecture

5.3 User Stories

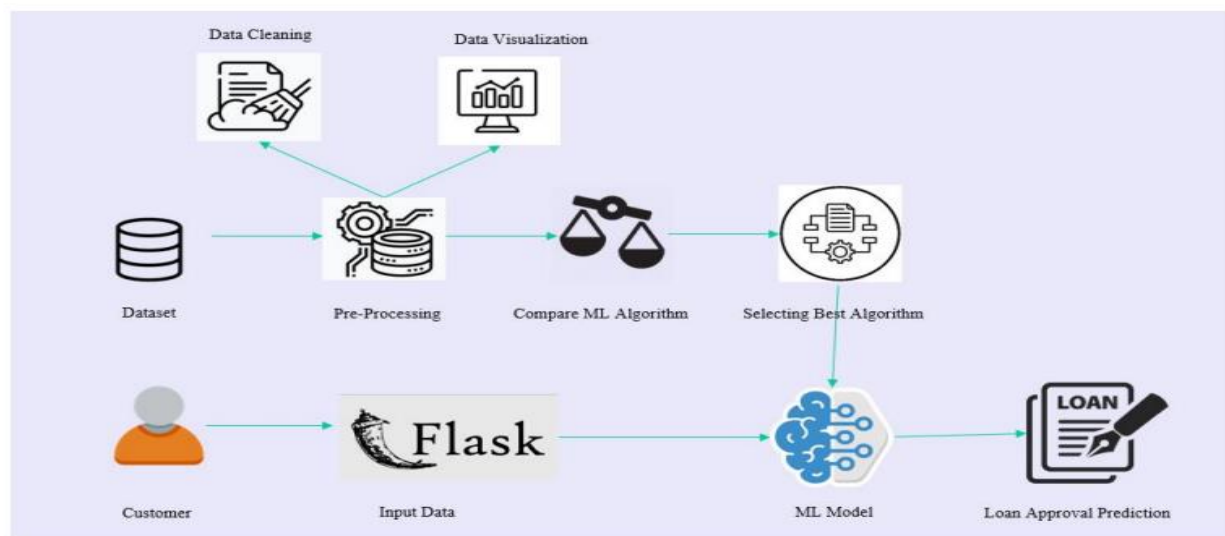
6. PROJECT PLANNING & SCHEDULING

6.1 Sprint

Planning &

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date(Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	8	6 Days	24 Oct 2022	17 Nov 2022	17 Nov 2022	11 Nov 2022
Sprint-2	10	6 Days	31 Oct 2022	17 Nov 2022	17 Nov 2022	10 Nov 2022
Sprint-3	13	6 Days	07 Nov 2022	17 Nov 2022	17 Nov 2022	29 Oct 2022
Sprint-4	13	6 Days	07 Nov 2022	17 Nov 2022	17 Nov 2022	28 Oct 2022

Estimation



6.2 Sprint Delivery Schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint – 1	Dashboard	USN – 1	As a user, I need to read the details given in the form.	2	High	2
Sprint – 2	Dashboard		As a user, after reading all the information I need to click on predict	2	Low	2
Sprint – 3	Details enter page	USN – 2	As a user, I need to enter all the required details in the corresponding fields.	2	Medium	2
Sprint – 3	Validation	USN – 3	Validates the truthfulness of entered information	1	High	1
Sprint – 4	Prediction	USN – 4	Prediction based on the details given by the user is done.	2	Medium	3
Sprint – 4	Display result	USN – 5	Final resulting page is displayed.	2	Low	4

6.3 Reports from JIRA

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date(Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	8	6 Days	24 Oct 2022	17 Nov 2022	17 Nov 2022	16 Nov 2022
Sprint-2	10	6 Days	31 Oct 2022	17 Nov 2022	17 Nov 2022	17 Nov 2022
Sprint-3	13	6 Days	07 Nov 2022	17 Nov 2022	17 Nov 2022	16 Nov 2022
Sprint-4	13	6 Days	07 Nov 2022	17 Nov 2022	17 Nov 2022	17 Nov 2022

Velocity: Imagine we have a 10-days print duration, and the velocity of the team is 20(points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

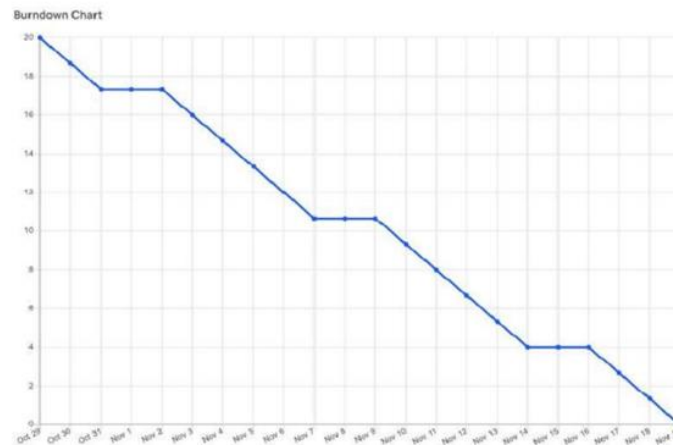
$$AV = \text{Velocity} / \text{Sprint duration}$$

Sprint	Average Velocity
Sprint1	1.33
Sprint2	1.67
Sprint3	2.17
Sprint4	2.17

Total Average Velocity = 1.83

Burnt Chart

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



7. CODING & SOLUTIONING

7.1 Feature 1

```
from flask import Flask, request, render_template, url_for
import numpy as np
import pickle
import requests
app = Flask(__name__)
@app.route('/')
def home():
    return render_template('Home.html',title = 'home',val = 'namaste')
@app.route('/detailPage')
def detail():
    return render_template('Details.html')
@app.route('/showResult',methods = ['POST'])
def result():
    gender,married,depend,education,self_emp,applicant_income,co_income,loan_amount,loan_term,credit_history,property_area = [x for x in request.form.values()]
    print("details",gender,married,depend,education,self_emp,applicant_income,co_income,loan_amount,loan_term,credit_history,property_area)

    if gender == 'Male':
        gender = 1
    else:
        gender = 0

    if married == 'Yes':
        married = 1
```

```

else:
    married = 0

if depend == '3+':
    depend = 3

if education == 'Graduate':
    education = 0
else:
    education = 1

if self_emp == 'Yes':
    self_emp = 1
else:
    self_emp = 0

applicant_income = int(applicant_income)
co_income = int(co_income)
loan_amount = int(loan_amount)
loan_term = int(loan_term)

if property_area == 'Urban':
    property_area = 2
elif property_area == 'Rural':
    property_area = 0
else:
    property_area = 1

features=[[gender,married,depend,education,self_emp,applicant_income,co_income,loan_amount,loan_
term,credit_history,property_area]]

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud
account.
API_KEY = "a1Ri3V6f6RR0LyqdgU7fIGlc0jfJaBeZ_Cqq34bQN4v6"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]
header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

# NOTE: manually define and pass the array(s) of values to be scored in the next line
payload_scoring={"input_data":[{"field":
[["gender","married","depend","education","self_emp","applicant_income","co_income","loan_amount
","loan_term","credit_history","property_area"]], "values": features}]}

```

```

response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/f4c91128-
dd2d-440c-85da-ab27eb9a0944/predictions?version=2022-11-15', json=payload_scoring,
headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
pred=response_scoring.json()
prediction=pred['predictions'][0]['values'][0][0]

#prediction=model.predict(features)
print(prediction)
#prediction = 1 # store the result value here
if(prediction == 0 ):
    return render_template('Fail.html')
else:
    return render_template('Success.html')

```

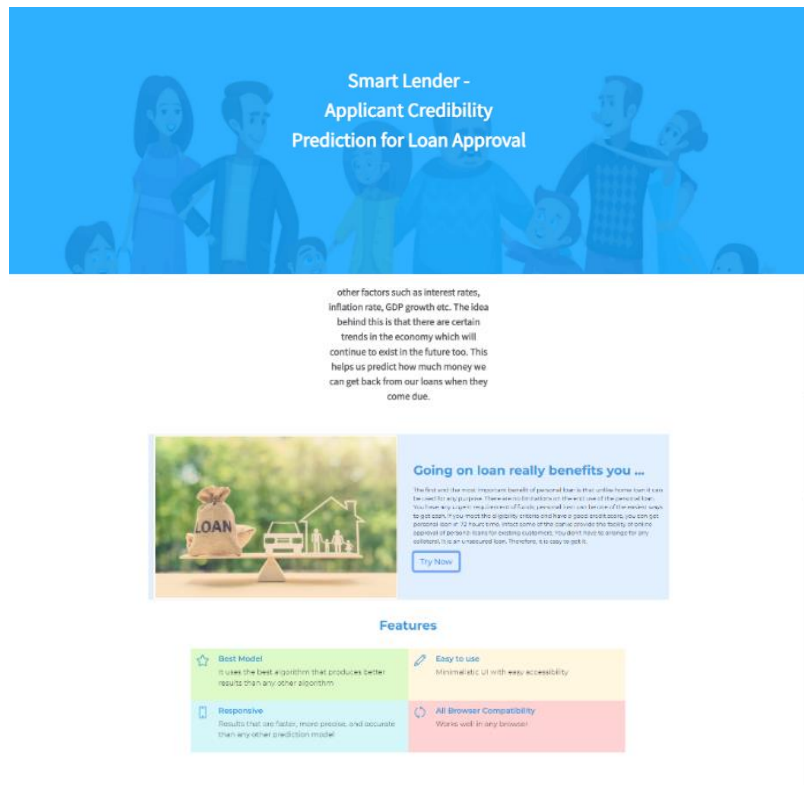
```

if __name__ == '__main__':
    app.run()

```

7.2 UI Design

HOMEPAGE



DETAIL GATHERING PAGE

LOAN ELIGIBILITY PREDICTION SYSTEM

FILL UP THE FOLLOWING DETAILS

GENDER	<input type="text" value="Male"/>	MARITAL STATUS	<input type="text" value="Married"/>
DEPENDENTS	<input type="text" value="2"/>	EDUCATION	<input type="text" value="Graduate"/>
SELF EMPLOYED	<input type="text" value="No"/>	APPLICANT INCOME	<input type="text" value="300000"/>
CO APPLICANT INCOME	<input type="text" value="100000"/>	LOAN AMOUNT	<input type="text" value="50000"/>
LOAN AMOUNT TERM (IN DAYS)	<input type="text" value="365"/>		
CREDIT HISTORY	<input type="text" value="1"/>	PROPERTY AREA	<input type="text" value="Urban"/>

Show Result

NOT ELIGIBLE

SORRY!!!
YOU ARE NOT ELIGIBLE FOR THE
LOAN TRY BY REDUCING THE QUOTED
LOAN AMOUNT

The result showed here is always an approximate result. Each bank follows different Loan sanction procedures. It only shows an approximate prediction through the information gathered from a trusted source

Give us your Feedback

Your feedback may help us to improve our model and correct our mistakes

<input type="text" value="Name"/>
<input type="text" value="Email"/>
<input type="text" value="Enter your feedback"/>

Send

Smart Lender © NaalayaThiranProject

ELIGIBLE

GREAT!!!
YOU ARE ELIGIBLE FOR THE LOAN

The result showed here is always an approximate result. Each bank follows different Loan sanction procedures. It only shows an approximate prediction through the information gathered from a trusted source

Give us your Feedback

Your feedback may help us to improve our model and correct our mistakes

<input type="text" value="Name"/>
<input type="text" value="Email"/>
<input type="text" value="Enter your feedback"/>

Send

Smart Lender © NaalayaThiranProject

8. TESTING

8.1 Test Cases

Test case ID	Feature Type	
TC_001	UI	
TC_002	Functional	
TC_003	UI	
TC_004	Functional	
TC_005	Functional	
TC_006	UI	
TC_007	Functional	
TC_008	UI	Re

DEFECT ANALYSIS

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Total
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won't Fix	1	0	1	0	2
Total	6	1	4	3	14

TEST CASE ANALYSIS

Section	Total Cases	Not Tested	Fail	Pass
Client Application	5	0	0	5
Security	1	0	0	1
Performance	2	0	0	2
Exception Reporting	1	0	0	1

	Status	Executed By
ected	PASS	Praveen S Anandha Kannan N
ected	PASS	Balaji T Hariharan C
ected	PASS	Praveen S Anandha Kannan N
ected	PASS	Balaji T Hariharan C
ected	PASS	Praveen S Anandha Kannan N
ected	PASS	Balaji T Hariharan C
xpected	PASS	Praveen S Anandha Kannan N
xpected	PASS	Balaji T Hariharan C

8.2 User Acceptance Testing

9. RESULTS

9.1 Performance Metrics

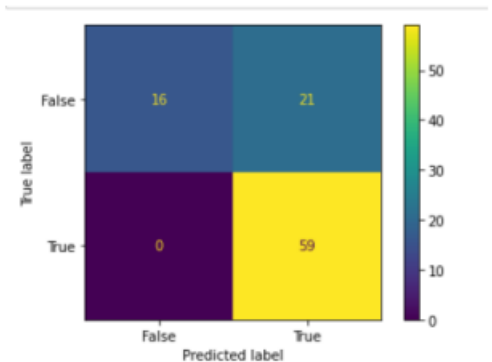
ACCURACY

CONTENT	VALUE
Training Accuracy	81.5%
Testing Accuracy	78.1%

CLASSIFICATION REPORT

CONTENT	VALUE
Precision	73.75%
Recall	100%
F1 Score	84.89%
Accuracy Score	78.125%

CONFUSION MATRIX



10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

- Easily identifies trends and patterns of Loan sanction style and criteria.
- Since, the whole system is automated , there will be no need of human intervention .
- Continuous Improvement of prediction by updating the training dataset with the current loan sanction trend dataset.

DISADVANTAGES

- The provided result cannot be always true.
- Periodic updation of model is required in order to maintain the model with current trend .

11. CONCLUSION

From a proper analysis of positive points and constraints on the component, it can be safely concluded that the product is a highly efficient component. This application is working properly and meeting to all Banker requirements. This component can be easily plugged in many other systems. Machine learning helps to understand the factors which affect the specific outcomes most. Other models like neural network and discriminate analysis can be used individually or combined for enhancing reliability and accuracy prediction.

12. FUTURE SCOPE

In near future this module of prediction can be integrate with the module of automated processing system. The system is trained on old training dataset in future software can be made such that new testing date should also take part in training data after some fix time.

13. APPENDIX

Source Code

```
import pandas as pd
import numpy as np
import pickle
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, f1_score
from sklearn.linear_model import LogisticRegression
data = pd.read_csv('C:\\Users\\rajaa\\naalaiya thiran\\dataset\\train_u6lujuX_CVtuZ9i.csv')
#Data preprocessing
data.drop(["Loan_ID"], axis=1, inplace=True)
#Handling Missing Values
def missing_values(df):
    a = num_null_values = df.isnull().sum()
    return a
# dropping the missing values
data = data.dropna()
#Encoding
from sklearn.preprocessing import OrdinalEncoder

ord_enc = OrdinalEncoder()
data[["Gender", 'Married', 'Dependents', 'Education', 'Self_Employed', 'Property_Area', 'Loan_Status']] =
ord_enc.fit_transform(data[["Gender", 'Married', 'Dependents', 'Education', 'Self_Employed', 'Property_Area', 'Loan
_Status']])
data.head()
#Train Test Split
X = data.drop("Loan_Status", axis=1)
y = data["Loan_Status"]
```

```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2, random_state=2)
print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)
```

```
#Naive bayes
```

```
from sklearn.naive_bayes import GaussianNB
gfc = GaussianNB()
gfc.fit(X_train, y_train)
pred1 = gfc.predict(X_test)
training_data_accuracy_nb = accuracy_score(pred1,y_test)
```

```
#XGB Classifier
```

```
from xgboost import XGBClassifier
```

```
xgb = XGBClassifier(learning_rate =0.1,
n_estimators=1000,
max_depth=3,
min_child_weight=1,
gamma=0,
subsample=0.8,
colsample_bytree=0.8,
objective= 'binary:logistic',
nthread=4,
scale_pos_weight=1,
seed=27)
xgb.fit(X_train, y_train)
pred3 = xgb.predict(X_test)
test_data_accuracy_xgb = accuracy_score(y_test,pred3)
```

```
#DecisionTreeClassifier
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import RandomizedSearchCV
```

```
def randomized_search(params, runs=20, clf=DecisionTreeClassifier(random_state=2)):
    rand_clf = RandomizedSearchCV(clf, params, n_iter=runs, cv=5, n_jobs=-1, random_state=2)
    rand_clf.fit(X_train, y_train)
    best_model = rand_clf.best_estimator_
```

```
# Extract best score
```

```
best_score = rand_clf.best_score_
```

```
# Print best score
```

```
print("Training score: {:.3f}".format(best_score))
```



```

# Predict test set labels
y_pred = best_model.predict(X_test)

# Compute accuracy
accuracy = accuracy_score(y_test, y_pred)

# Print accuracy
print('Test score: {:.3f}'.format(accuracy))

return best_model

#RandomForest

from sklearn.ensemble import RandomForestClassifier
randomized_search(params={
    'min_samples_leaf':[1,2,4,6,8,10,20,30],
    'min_impurity_decrease':[0.0, 0.01, 0.05, 0.10, 0.15, 0.2],
    'max_features':['auto', 0.8, 0.7, 0.6, 0.5, 0.4],
    'max_depth':[None,2,4,6,8,10,20],
}, clf=RandomForestClassifier(random_state=2))

#Knn
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=7)
knn.fit(X_train, y_train)
predictions = knn.predict(X_test)
test_data_accuracy_knn = accuracy_score(y_test,predictions)

#IBM Watson
from ibm_watson_machine_learning import APIClient
import json
import numpy as np
wml_credentials = {
    "url":"https://us-south.ml.cloud.ibm.com",
    "apikey":"YFDhnC_5-8fBFh1-nDJtM9uQsD7DyDCg64GRz2MhOdI6"
}
wml_client = APIClient(wml_credentials)

```

Feature 2

APP.PY :-

```

from flask import Flask, request, render_template, url_for
import numpy as np
import pickle

```

```

import requests
app = Flask(__name__)
@app.route('/')
def home():
    return render_template('Home.html',title = 'home',val = 'namaste')
@app.route('/detailPage')
def detail():
    return render_template('Details.html')
@app.route('/showResult',methods = ['POST'])
def result():
    gender,married,depend,education,self_emp,applicant_income,co_income,loan_amount,loan_term,credit_history,property_area = [x for x in request.form.values()]
    print("details",gender,married,depend,education,self_emp,applicant_income,co_income,loan_amount,loan_term,credit_history,property_area)

    if gender == 'Male':
        gender = 1
    else:
        gender = 0

    if married == 'Yes':
        married = 1
    else:
        married = 0

    if depend == '3+':
        depend = 3

    if education == 'Graduate':
        education = 0
    else:
        education = 1

    if self_emp == 'Yes':
        self_emp = 1
    else:
        self_emp = 0

    applicant_income = int(applicant_income)
    co_income = int(co_income)
    loan_amount = int(loan_amount)
    loan_term = int(loan_term)

    if property_area == 'Urban':
        property_area = 2
    elif property_area == 'Rural':

```

```

        property_area = 0
    else:
        property_area = 1

features=[[gender,married,depend,education,self_emp,applicant_income,co_income,loan_amount,loan_
term,credit_history,property_area]]

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud
account.
API_KEY = "a1Ri3V6f6RR0LyqdgU7fIGlc0jfJaBeZ_Cq34bQN4v6"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]
header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

# NOTE: manually define and pass the array(s) of values to be scored in the next line
payload_scoring={"input_data":[{"field":
[["gender","married","depend","education","self_emp","applicant_income","co_income","loan_amount
","loan_term","credit_history","property_area"]], "values": features}]}

response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/f4c91128-
dd2d-440c-85da-ab27eb9a0944/predictions?version=2022-11-15', json=payload_scoring,
headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
pred=response_scoring.json()
prediction=pred['predictions'][0]['values'][0][0]

#prediction=model.predict(features)
print(prediction)
#prediction = 1 # store the result value here
if(prediction == 0):
    return render_template('Fail.html')
else:
    return render_template('Success.html')

if __name__ == '__main__':
    app.run()

```

GitHub Link

<https://github.com/IBM-EPBL/IBM-Project-2258-1658468344>

ProjectDemoLink

https://drive.google.com/file/d/19U-_Gg-SABpU0Gz4fzNimOvwmyRIRjlb/view?usp=share_link